International Journal of Mathematics and
Computer Science, **16**(2021), no. 1, 487–502

Modification of the finite-difference method for solving a special class of nonlinear two-point boundary value problems

Ivo Beroš[1], Nikica Hlupić[2], Danko Basch[2]

[1]Institute for Tourism
Vrhovec 5
Zagreb, Croatia

[2]Faculty of Electrical Engineering
University of Zagreb
Zagreb, Croatia

email: ivo.beros@iztzg.hr, nikica.hlupic@fer.hr, danko.basch@fer.hr

(Received September 2, 2020, Accepted October 6, 2020)

Abstract

The finite-difference method for solving a nonlinear two-point boundary value problems consists of discretization and solving the resulting system of nonlinear equations, usually by the Newton's method. In this paper, we modify Newton's method to solve the system of nonlinear equations related to the discretization of a special class of the nonlinear two-point boundary value problems, where y'' is expressed as a multivariate polynomial in y and y'. In the proposed method, the new iteration point is determined as the minimal point of the square of 2-norm of residuals along Newton direction. The minimal point is computed using the solution of a univariate polynomial equation. The numerical examples show that the proposed method performs better than the standard Newton's method with the unit step length.

Key words and phrases: System of nonlinear equations, Boundary value problems, System of polynomial equations.
AMS (MOS) Subject Classifications: 65L10, 65L12, 65H10.
ISSN 1814-0432, 2021, http://ijmcs.future-in-tech.net

1 Introduction

This article improves the standard finite-difference method for numerical solution of the two-point nonlinear boundary value problem.

Two-point boundary value problems occur as mathematical models in many branches of applied sciences and engineering. However, in many cases, the boundary value problems cannot be solved analytically. Therefore, it is necessary to consider various numerical methods to obtain an approximation of the solutions.

The most common numerical methods for solving a two-point nonlinear boundary value problem are the shooting method [3, 13], the finite-difference methods [3, 7, 10, 13, 15], the monotone iterative methods ([5, 8]), the Adomian decomposition method [1, 9, 16], the quasilinearization method [4, 14] and the collocation methods [2]. An interesting survey of numerical methods for solving nonlinear two-point boundary value problem can be found in [6].

We study two-point boundary value problems of the form

$$y''(x) = F(x, y, y'), \quad a < x < b, \tag{1.1}$$

subject to the boundary conditions

$$y(a) = y_a, \quad y(b) = y_b. \tag{1.2}$$

We suppose that the boundary value problem (1.1)-(1.2) has unique solution. More details about the question of the existence and uniqueness of the solution of problem (1.1)-(1.2) can be found in [3, 13].

Moreover, function F is continuous on the set $D = \{(x, u, v)|a \leq x \leq b, u, v \in \mathbb{R}\}$. Consider the case when F can be written as

$$F(x, u, v) = \sum_i f_i(x) u^{r_i} v^{s_i}, \qquad r_i, s_i \in \mathbb{N}_0; \tag{1.3}$$

i.e. y'' can be expressed as a multivariate polynomial in y and y'.

The common finite-difference method consists of discretization of boundary value problem and solving the resulting system of nonlinear equations, usually by the Newton's method with unit step length. We propose a modification of the Newton's method by computing the new iteration point as the minimal point of the square of the 2-norm of residuals along the Newton direction. This approach is suitable for the boundary value problems where F takes form in (1.3) because the minimal point can be determined by solving a univariate polynomial equation. The proposed method leads to better accuracy and faster convergence.

2 Modified finite-difference method

The first step in the common finite-difference methods for solving a problem (1.1)-(1.2) is discretization. The interval $[a, b]$ is divided into $N + 1$ equal subintervals $[x_{i-1}, x_i]$, $i = 1, \ldots, N + 1$, where N is positive integer, $x_i = a + ih$, and $h = (b - a)/(N + 1)$. Let y_i denote an approximate solution value at the mesh points x_i. In the mesh points x_i, derivatives $y''(x_i)$ and $y'(x_i)$ are replaced with difference quotients,

$$\begin{aligned} y''(x_i) &= \frac{y_{i-1} - 2y_i + y_{i+1}}{h^2} + O(h^2), \\ y'(x_i) &= \frac{y_{i+1} - y_{i-1}}{2h} + O(h^2), \end{aligned} \qquad i = 1, \ldots, N. \tag{2.4}$$

Using (2.4), we can approximate (1.1) at $x = x_i$ by

$$\frac{y_{i-1} - 2y_i + y_{i+1}}{h^2} - F\left(x_i, y_i, \frac{y_{i+1} - y_{i-1}}{2h}\right) = 0, \quad 1 \le i \le N. \tag{2.5}$$

Boundary conditions (1.2) give $y_0 = y_a$ and $y_{N+1} = y_b$.

The resulting system of nonlinear equations (2.5) can be written as $\mathbf{G}(\mathbf{y}) = \mathbf{0}$, where $\mathbf{G} : \mathbb{R}^N \to \mathbb{R}^N$, $\mathbf{G}(\mathbf{y}) = (g_1(\mathbf{y}), \ldots, g_N(\mathbf{y}))$, $\mathbf{y} = (y_1, \ldots, y_N)$ and functions $g_i : \mathbb{R}^N \to \mathbb{R}$ (residuals) are defined by

$$g_i(\mathbf{y}) = y_{i-1} - 2y_i + y_{i+1} - h^2 F\left(x_i, y_i, \frac{y_{i+1} - y_{i-1}}{2h}\right), \quad 1 \le i \le N. \tag{2.6}$$

For convenience, we imply that $y_0 = y_a$ and $y_{N+1} = y_b$.

The second step is the solving of the system $\mathbf{G}(\mathbf{y}) = \mathbf{0}$. We assume that the ordinary Newton's method is used. Given a initial solution $\mathbf{y}^{(0)}$, a sequence of approximations $\mathbf{y}^{(1)}$, $\mathbf{y}^{(2)}, \ldots$ to the solution of $\mathbf{G}(\mathbf{y}) = \mathbf{0}$ is generated by $\mathbf{y}^{(i+1)} = \mathbf{y}^{(i)} + \mathbf{s}^{(i)}$, where $\mathbf{s}^{(i)}$ (the Newton direction) is the solution of the linear system $\mathbf{J}(\mathbf{y}^{(i)})\mathbf{s}^{(i)} = -\mathbf{G}(\mathbf{y}^{(i)})$ and $\mathbf{J}(\mathbf{y}^{(i)})$ is the Jacobian matrix of $\mathbf{G}$.

In the rest of the paper, by the term *standard finite-difference* method we mean the finite-difference method where y'' and y' are expressed as in (2.4) and the nonlinear system $\mathbf{G}(\mathbf{y}) = \mathbf{0}$ is solved using the common Newton's method.

Now we exploit special structure (1.3) of F. Since F is multivariate polynomial in the last two variables, functions g_i are also multivariate polynomials. More precisely, functions g_i are multivariate polynomials in three variables: y_{i-1}, y_i and y_{i+1}. Therefore, system $\mathbf{G}(\mathbf{y}) = \mathbf{0}$ is a system of polynomial equations. The degree of polynomial g_i is $d_M = \max\{1, \max_j\{r_j + s_j\}\}$.

Following [11, 12], to solve system $\mathbf{G}(\mathbf{y}) = \mathbf{0}$, we evaluate the new iteration point as the minimal point of the square of 2-norm of residuals along the Newton direction. The new iteration point is computed by solving a univariate polynomial equation. The algorithm for solving system $\mathbf{G}(\mathbf{y}) = \mathbf{0}$ is

1. Let $\mathbf{y}^{(0)}$ is an initial approximation of solution.

2. Evaluate Newton direction $\mathbf{s}^{(i)}$ by solving the linear system

$$\mathbf{J}(\mathbf{y}^{(i)})\mathbf{s}^{(i)} = -\mathbf{G}(\mathbf{y}^{(i)}),$$

3. Let λ^* be the minimal point of the function $RSS : \mathbb{R} \to \mathbb{R}$, where RSS is square of 2-norm of residuals along the Newton direction, [3] defined by

$$RSS(\lambda) = \frac{1}{2}\|\mathbf{G}(\mathbf{y} + \lambda\mathbf{s})\|_2^2 = \frac{1}{2}\sum_{i=1}^{n} g_i^2(\mathbf{y} + \lambda\mathbf{s}). \tag{2.7}$$

4. A new approximation is $\mathbf{y}^{(i+1)} = \mathbf{y}^{(i)} + \lambda^*\mathbf{s}^{(i)}$.

5. If the stopping criteria is not satisfied, then go to step 2. Otherwise, terminate the iteration process.

We now describe how to form the function $RSS(\lambda)$, needed for calculation of λ^*. First, note that the matrix $\mathbf{J}(\mathbf{y})$ (for the purpose of simplicity, we write $\mathbf{y}$, $\mathbf{s}$ instead $\mathbf{y}^{(i)}$, $\mathbf{s}^{(i)}$ respectively) is the three diagonal Jacobian matrix

$$\mathbf{J}(\mathbf{y}) = \frac{\partial \mathbf{G}}{\partial \mathbf{y}} = \begin{pmatrix} b_1 & c_1 & 0 & \dots & 0 \\ a_2 & b_2 & c_2 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & a_{n-1} & b_{n-1} & c_{n-1} \\ 0 & \dots & 0 & a_n & b_n \end{pmatrix} \tag{2.8}$$

with elements defined by

$$\begin{aligned} a_i = \frac{\partial g_i(\mathbf{y})}{\partial y_{i-1}} = &\quad 1 + \frac{h}{2}\frac{\partial F}{\partial v}\left(x_i, y_i, \frac{y_{i+1}-y_{i-1}}{2h}\right) \\ b_i = \frac{\partial g_i(\mathbf{y})}{\partial y_i} = &-2 + h^2\frac{\partial F}{\partial u}\left(x_i, y_i, \frac{y_{i+1}-y_{i-1}}{2h}\right) \\ c_i = \frac{\partial g_i(\mathbf{y})}{\partial y_{i+1}} = &\quad 1 - \frac{h}{2}\frac{\partial F}{\partial v}\left(x_i, y_i, \frac{y_{i+1}-y_{i-1}}{2h}\right). \end{aligned} \tag{2.9}$$

Determining $\lambda^* = \min_{\lambda\in\mathbb{R}} RSS(\lambda)$ in a general case is too expensive. However, in this case it is relatively effortless to compute λ^* because $g_i(\mathbf{y}+\lambda\mathbf{s})$ are polynomials of degree d_M. Thus $RSS(\lambda)$ is polynomial of degree $2d_M$ and λ^* is obtained by solving polynomial equation $RSS'(\lambda) = 0$.

Since the function $RSS(\lambda)$ is polynomial, its Taylor series is

$$RSS(\lambda) = \frac{1}{2}\left(r_0 + r_1\lambda + r_2\frac{\lambda^2}{2!} + \cdots + r_{2d_M}\frac{\lambda^{2d_M}}{(2d_M)!}\right). \tag{2.10}$$

To determine the coefficients $r_0, \ldots, r_{2d_M}$, we need to introduce the auxiliary functions $\phi_i(\lambda) = g_i(\mathbf{y}+\lambda\mathbf{s})$ and $RSS_i(\lambda) = \phi_i^2(\lambda)/2$. Obviously, RSS is sum of RSS_i and the coefficients r_j in (2.10) are the sum of coefficients of the Taylor series for RSS_i.

Since $RSS_i(\lambda)$ is also polynomial of degree $2d_M$, the Taylor series for $RSS_i(\lambda)$ is

$$RSS_i(\lambda) = \frac{1}{2}\left(\phi_i^2(0) + (\phi_i^2)'(0)\lambda + \cdots + (\phi_i^2)^{(2d_M)}(0)\frac{\lambda^{2d_M}}{(2d_M)!}\right). \tag{2.11}$$

By the product rule, high-order derivations of ϕ^2 can be evaluated as

$$(\phi_i^2)^{(m)} = \sum_{j=0}^{m}\binom{m}{j}\phi_i^{(j)}\phi_i^{(m-j)}.$$

Since the degree of $\phi_i(\lambda)$ is d_M, it is easy to see that

$$(\phi_i^2)^{(m)}(0) = \sum_{j=\max\{0,m-d_M\}}^{\min\{d_M,m\}} \binom{m}{j} \phi_i^{(j)}(0)\phi_i^{(m-j)}(0). \tag{2.12}$$

From (2.12) follows that all coefficients in (2.11) can be evaluated from the values $\phi_i^{(j)}(0)$, $j = 0, \ldots, d_M$.

The $\phi_i(0)$ is already evaluated, $\phi_i(0) = g_i(\mathbf{y}) = (\mathbf{G}(\mathbf{y}))_i$. Moreover, by the definition

$$\begin{aligned}\phi_i'(\lambda) &= \frac{d}{d\lambda} g_i(\mathbf{y} + \lambda\mathbf{s}) = \sum_{j=1}^{N} \frac{\partial g_i(\mathbf{y} + \lambda\mathbf{s})}{\partial y_j} \frac{d(\mathbf{y} + \lambda\mathbf{s})_j}{d\lambda} \\ &= \sum_{j=1}^{N} \frac{\partial g_i(\mathbf{y} + \lambda\mathbf{s})}{\partial y_j} s_j. \end{aligned} \tag{2.13}$$

Since $\partial g_i / \partial dy_j = 0$, $j \notin \{i-1, i, i+1\}$, (2.13) and (2.9),

$$\begin{aligned}\phi_i'(0) &= \frac{\partial g_{i-1}(\mathbf{y})}{\partial y_{i-1}} s_{i-1} + \frac{\partial g_i(\mathbf{y})}{\partial y_i} s_i + \frac{\partial g_{i+1}(\mathbf{y})}{\partial y_{i+1}} s_{i+1} \\ &= a_i s_{i-1} + b_i s_i + c_i s_{i+1}, \end{aligned} \tag{2.14}$$

where we use $s_0 = s_{N+1} = 0$. Similarly,

$$\phi_i''(0) = \frac{d}{d\lambda} \sum_{j=1}^{N} \frac{\partial g_i(\mathbf{y})}{\partial y_j} s_j = \sum_{j=i-1}^{i+1} \sum_{k=i-1}^{i+1} \frac{\partial^2 g_i(\mathbf{y})}{\partial y_k \partial y_j} s_j s_k.$$

The higher-order derivations are calculated in an analogous way. Theoretically, the number of partial derivatives needed for calculation grows up exponentially, but in the practical situations many of them vanishes quickly.

In cases where it is known how to find Taylor series for RSS_i, it is easy to calculate coefficients r_j in (2.10) and determine $RSS'(\lambda)$,

$$RSS'(\lambda) = \frac{1}{2}\left(r_1 + r_2\lambda + r_3\frac{\lambda^2}{2!} + \cdots + r_{2d_M}\frac{\lambda^{2d_M-1}}{(2d_M-1)!} \right). \tag{2.15}$$

The step length λ^* is solution of equation $RSS'(\lambda) = 0$ and we use the standard univariate Newton's method with $\lambda_0 = 1$ to solve it.

3 Numerical examples

For numerical testing we consider several nonlinear two-point boundary value problems where F can be written as a polynomial in y and y'. In examples E1, E1a, E2, E2a, and E3 below, a function F has the form

$$F(x, y, y') = \sum_i \alpha_i(x) y + \sum_j \beta_j(x) y',$$

while examples E4 and E4a contain term $y^\alpha y'^\beta$, $\alpha, \beta \geq 1$.

E1. $y''(x) = -\frac{2}{x}y'(x) - y^5(x)$, $0 < x < 1$, boundary conditions $y(0) = 1$, $y(1) = \sqrt{3}/2$, and exact solution $y(x) = \frac{1}{\sqrt{1 + x^2/3}}$ [9].

E1a. $y''(x) = -\frac{2}{x}y'(x) - y^5(x)$, $0 < x < 2$, boundary conditions $y(0) = 1$, $y(2) = \sqrt{3}/\sqrt{7}$, and exact solution $y(x) = \frac{1}{\sqrt{1 + x^2/3}}$.

E2. $y''(x) = y^3(x) - 6y(x) - 2x^3$, $1 < x < 2$, boundary conditions $y(1) = 2$, $y(2) = 2.5$, and exact solution $y(x) = x + 1/x$ [10].

E2a. $y''(x) = y^3(x) - 6y(x) + 6x + 6x^3 - x^9$, $1 < x < 4$, boundary conditions $y(1) = 1$, $y(2) = 64$, and exact solution $y(x) = x^3$.

E3. $y''(x) = -(1 + y'(x)^2)$, $0 < x < 1$, boundary conditions $y(0) = 0$, $y(1) = 0$, and exact solution $y(x) = \ln\left(\frac{\cos(x - 1/2)}{\cos(1/2)}\right)$ [6].

E4. $y''(x) = \dfrac{32 + 2x^3 - yy'}{8}$, $1 < x < 3$, boundary conditions $y(1) = 17$, $y(3) = 43/3$, and exact solution $y(x) = x^2 + 16/x$ [10].

E4a. $y''(x) = \dfrac{32 + 2x^3 - yy'}{8}$, $1 < x < 5$, boundary conditions $y(1) = 17$, $y(3) = 28.2$, and exact solution $y(x) = x^2 + 16/x$.

For each problem, we compare the standard and the proposed finite-difference method on the meshes with $N = 15$, 63, and 255, respectively. For each iteration of the Newton's method we present a maximum residual error ($\max_i |\mathbf{G}(\mathbf{y})|$) for the standard method (E), maximum residual error for the proposed method (E_M), and R, ratio between E and E_M. The cases when the ratio R is greater than 100 are marked ***. An initial approximation of the solution is given by $\mathbf{y}^{(0)} = (y_1^{(0)}, \ldots, y_N^{(0)})$, where $y_i^{(0)} = y_a + i \cdot \frac{y_b - y_a}{N+1}$. All calculations were performed in the Windows 10 operating system in the Octave environment, version 5.2.0.

For problem E1 only, we show how to evaluate the coefficients of the polynomials $RSS_i(\lambda)$. For the other problems, the coefficients are evaluated in a similar manner.

In problem E1, the function F is given by $F(x, u, v) = -\frac{2}{x}v - u^5$ which is clearly a polynomial in the variables u and v. The functions g_i take the form

$$g_i(y_{i-1}, y_i, y_{i+1}) = y_{i-1} - 2y_i + y_{i+1} + h^2 \left(\frac{y_{i+1} - y_{i-1}}{x_i h} + y_i^5(x) \right), i = 1, \ldots, n.$$

The first-order partial derivatives of g_i are

$$\frac{\partial g_i}{\partial y_{i-1}} = 1 - \frac{h}{x_i}, \qquad \frac{\partial g_i}{\partial y_{i+1}} = 1 + \frac{h}{x_i}, \qquad \frac{\partial g_i}{\partial y_i} = -2 + 5\, h^2 y_i^4.$$

Table 1: Maximum residual errors for Example 1

	$N=15$			$N=31$			$N=63$		
k	E	E_M	R	E	E_M	R	E	E_M	R
1	2.5-05	2.5-05	1.00	1.5-06	1.6-06	1.00	9.7-08	9.7-08	1.00
2	6.8-08	6.7-08	1.02	4.1-09	4.2-09	0.98	2.6-10	2.6-10	0.98
3	2.0-13	6.0-15	32.98	1.1-14	4.8-16	24.02	8.5-16	2.2-16	3.94

There are only four remaining non-trivial high-order partial derivatives

$$\frac{\partial^k g_i}{\partial y_i^k} = h^2 \frac{5!}{(5-k)!} y_i^{5-k}, \quad k = 2, \ldots, 5.$$

For given $\mathbf{y}$, $\mathbf{s}$, values $\phi_i^{(j)}(0)$, $j = 0, \ldots, 5$, are

$$\begin{aligned}
\phi_i(0) &= y_{i-1} - 2y_i + y_{i+1} + h^2 \left(\frac{y_{i+1} - y_{i-1}}{x_i h} + y_i^5(x) \right) \\
\phi_i'(0) &= \left(1 - \frac{h}{x_i}\right) s_{i-1} + \left(-2 + 5\, h^2 y_i^4\right) s_i + \left(1 + \frac{h}{x_i}\right) s_{i+1} \\
\phi_i^{(k)}(0) &= h^2 \frac{5!}{(5-k)!} y_i^{5-k} s_i^k, \quad k = 2, \ldots, 5.
\end{aligned}$$

The coefficients of the polynomials $RSS_i(\lambda)$ are calculated by use of (2.11) and (2.12).

In Table 1 the results of numerical experiment are presented. As we can see, the significant difference between standard and proposed method exists in the third iteration.

Problem E1a is a modification of the problem E1. The same differential equation is consider. However, for $0 < x < 2$. Consequently, the boundary conditions are $y(0) = 1$, $y(2) = \sqrt{3}/\sqrt{7}$. The results of the experiment show that the proposed method clearly outperformed the standard method (Table 2).

Problems E2 and E2a are examples of boundary value problems, where

Table 2: Maximum residual errors for Example 1a

	$N = 15$			$N = 31$			$N = 63$		
k	E	E_M	R	E	E_M	R	E	E_M	R
1	3.6-02	7.5-03	4.74	6.3-03	3.6-03	1.76	4.3-04	8.2-04	0.52
2	1.8-02	2.2-03	8.02	9.1-04	5.0-04	1.81	6.2-05	7.1-05	0.88
3	1.4-03	8.2-04	1.68	1.1-02	1.5-04	71.00	5.4-04	1.2-05	44.80
4	7.5-04	8.2-06	91.96	5.6-04	9.9-06	57.22	2.4-05	4.4-06	5.36
5	7.7-05	8.5-08	***	1.4-04	1.1-07	***	7.6-06	5.1-07	14.93
6	2.8-06	8.1-14	***	1.2-05	6.7-11	***	5.7-07	2.4-09	***
7				1.1-06	1.7-16	***	4.6-08	1.4-13	***
8							2.1-10	1.9-16	***

Table 3: Maximum residual errors for Example 2

	$N = 15$			$N = 31$			$N = 63$		
k	E	E_M	R	E	E_M	R	E	E_M	R
1	9.7-03	6.0-03	1.62	6.1-04	4.3-04	1.43	3.8-05	2.7-05	1.41
2	2.4-04	6.9-06	34.19	1.5-05	5.7-07	26.44	9.4-07	3.8-08	24.79
3	1.3-07	5.3-11	***	8.4-09	6.2-12	***	5.3-10	4.4-13	***
4	3.9-14	6.0-16	64.98	3.0-15	6.5-16	4.59	9.9-16	7.7-16	1.29

y' is not present. They have the same polynomial structure and different exact solutions. The results presented in Table 3 and Table 4 show that the proposed method is better again. The difference between the methods is more noticeable in the case of problem E2a, where the proposed method needs one step less than the standard method to achieve the desired accuracy.

The problem E3 represents a case when y is not present. Again, the proposed method achieves better results compared to the standard method (Table 5).

In the case when function F contains the term $y^{\alpha}y'^{\beta}$, $\alpha, \beta \geq 1$, the proposed method is better than the standard method when the initial solution

Table 4: Maximum residual errors for Example 2a

	$N = 15$			$N = 31$			$N = 63$		
k	E	E_M	R	E	E_M	R	E	E_M	R
1	5.0+02	2.7+02	1.88	3.2+01	1.7+01	1.91	2.0+00	1.0+00	1.91
2	6.8+01	2.0+01	3.34	4.3+00	1.3+00	3.35	2.7-01	8.1-02	3.35
3	6.4+00	1.2+00	5.37	4.0-01	7.6-02	5.26	2.5-02	4.8-03	5.25
4	3.4-01	2.8-02	12.52	2.2-02	1.8-03	11.84	1.3-03	1.1-04	11.79
5	5.1-03	4.3-05	***	3.3-04	2.7-06	***	2.1-05	1.7-07	***
6	3.0-06	1.3-10	***	1.8-07	7.7-12	***	1.1-08	4.8-13	***
7	1.1-12	0.0+00	***	6.3-14	1.1-13	0.56	7.1-15	8.0-15	0.89

Table 5: Maximum residual errors for Example 3

	$N = 15$			$N = 31$			$N = 63$		
k	E	E_M	R	E	E_M	R	E	E_M	R
1	7.5-04	5.8-04	1.29	5.7-05	4.8-05	1.21	3.8-06	3.2-06	1.19
2	3.7-06	6.0-07	6.12	4.2-07	5.0-08	8.52	3.1-08	3.3-09	9.31
3	1.6-11	6.3-12	2.59	3.7-12	8.2-13	4.49	3.2-13	5.9-14	5.44
4	4.0-17	3.6-17	1.12	5.6-17	3.4-17	1.64	4.7-17	3.6-17	1.29

is not close to the real solution of the problem. Namely, in problems E4 and E4a differential equations are the same, but the example E4 is valid for $1 < x < 3$, while E4a is valid for $1 < x < 5$. The results from Table 6 and Table 7 show that the standard method is better for solving problem E4, but the proposed method is better suited for problem E4a.

The possible solution of that problem is the incorporation of the additional conditions in the proposed method, such as Wolfe conditions, to achieve better properties of the method.

Table 6: Maximum residual errors for the problem E4

	$N = 15$			$N = 31$			$N = 63$		
k	E	E_M	R	E	E_M	R	E	E_M	R
1	2.9-02	2.8-02	1.06	1.8-03	1.7-03	1.05	1.1-04	1.1-04	1.05
2	8.7-05	1.1-04	0.76	5.6-06	7.2-06	0.78	3.5-07	4.5-07	0.79
3	6.2-10	1.2-09	0.53	4.1-11	7.2-11	0.56	2.6-12	4.5-12	0.57
4	2.6-15	2.6-15	1.00	2.9-15	2.9-15	1.00	3.1-15	3.1-15	1.00

Table 7: Maximum residual errors for the problem E4a

	$N = 15$			$N = 31$			$N = 63$		
k	E	E_M	R	E	E_M	R	E	E_M	R
1	2.9-01	2.9-01	0.97	1.7-02	1.8-02	0.96	1.1-03	1.1-03	0.96
2	9.1-03	8.1-03	1.13	5.7-04	4.6-04	1.24	3.5-05	2.9-05	1.24
3	1.6-05	5.5-06	2.97	9.7-07	2.9-07	3.38	6.1-08	1.8-08	3.38
4	2.9-11	1.3-12	22.68	1.8-12	6.4-14	27.48	1.1-13	7.5-15	14.74
5	5.8-15	4.9-15	1.18	7.1-15	6.0-15	1.17	5.7-15	5.6-15	1.02

4 Conclusion

In this paper, we considered the finite-difference method for solving the special class of the nonlinear two-point boundary value problems in which y'' is expressed as a multivariate polynomial in y and y'. We showed that the system of nonlinear equations arising from the problem is the system of polynomial equations and to solve the system a modified Newton's method was proposed, where the new iteration point is determined as the minimal point of the square of 2-norm of residuals along the Newton direction. The minimal point was determined by solving a univariate polynomial equation. The proposed method was tested on several examples in the literature. Based on the test results, we concluded that the proposed method achieved the desired accuracy faster than the standard method. This was especially no-

ticeable when F does not contain term $y^{\alpha}y'^{\beta}$, $\alpha, \beta \geq 1$, and when the initial approximation of the solution of the system of polynomial equations differed significantly from the solution.

References

[1] G. Adomian, Solving Frontier problems of Physics: The Decomposition Method, Kluwer, 1994.

[2] U. M. Ascher, J. Christiansen, R.D. Russell, Collocation software for boundary-value ODEs, ACM T Math. Software, **7**, no. 2, (1981), 209–222.

[3] U. M. Ascher, R. M. M. Mattheij, R. D. Russell, Numerical Solution of Boundary Value Problems for Ordinary Differential Equations, SIAM, 1995.

[4] R. E. Bellman, R. E. Kalaba, Quasilinearization and nonlinear boundary-value problems, Elsevir, 1965.

[5] M. Cherpion, C. De Coster, P. Habets, A constructive monotone iterative method for second-order BVP in the presence of lower and upper solutions, Appl. Math. Comput., **123**, no. 1, (2001), 75–91.

[6] S. Cuomo, A. Marasco, A numerical approach to nonlinear two-point boundary value problems for ODEs, Comput. Math. Appl, **55**, no. 11, (2008), 2476–2489.

[7] E. M. E. Elbarbary, M. El-Kady, Chebyshev finite difference approximation for the boundary value problems, Appl. Math. Comput., **139**, (2003), 513–523.

[8] P. W. Eloe, Y. Zhang, A quadratic monotone iteration scheme for two-point boundary value problems for ordinary differential equations, Nonlinear Analysis, **33**, no. 5, (1998), 443–453.

[9] F. Geng, M. Cui, A novel method for nonlinear two-point boundary value problems: Combination of ADM and RKM. Appl. Math. Comput., **217**, no. 9, (2011), 4676–4681.

[10] S.N. Ha, A nonlinear shooting method for two-point boundary value problems, Comput. Math. Appl, **42**, no. 10, (2001), 1411–1420.

[11] N. Hlupić, I. Beroš, D. Basch, A derivative-free algorithm for finding least squares solutions of quasi-linear and linear systems, CIT, **21**, no. 2, (2013), 125–135.

[12] N. Hlupic, I. Beroš, Solving Polynomial Systems by Penetrating Gradient Algorithm Applying Deepest Descent Strategy, arXiv preprint, arXiv:1501.03341, (2015).

[13] H. B. Keller, Numerical Methods for Two-Point Boundary-Value Problems, Dover Publications, 2018.

[14] R. A. Khan, The generalized quasi linearization technique for a second order differential equation with separated boundary conditions, Math. Comput. Modelling, **43**, no. 7–8, (2006), 727–742.

[15] I. A. Tirmizi, E. H. Twizell, Higher-order finite difference methods for nonlinear second-order two-point boundary-value problems, Appl. Math. Lett., **15**, (2002), 897–902.

[16] A. M. Wazwaz, A new algorithm for calculating Adomian polynomials for nonlinear operators, Appl. Math. Comput., **111,** (2000), 33–51.